Chapter 4
MATLAB Fundamentals

MATLAB (Matrix Laboratory) [150] is a high performance numerical computing environment and fourth generation programming language developed by Mathworks. It provides an interactive environment with loads of built in functions for matrix manipulation, technical computation, graphics and animation. Using MATLAB, mathematical computations can be easily performed. MATLAB can be used to analyze data, develop versatile range of applications and create models. Visualization as well as application development can also be performed using MATLAB. Interactive applications can also be developed using MATLAB. MATLAB is better than traditional applications like spreadsheets as well as programming languages because it provides advanced tools for the computational and graphical applications. It also provides built in functions for many applications, which need to be coded in traditional programming languages. MATLAB is used in wide range of applications like signal processing and communications, image and video processing, control systems, test and measurement, computational finance and computational biology. MATLAB is used by the researchers, scientists, engineers and academicians in industry, research centers and academic institutions for many applications like pattern recognition, signal processing etc. [156].

4.1Starting MATLAB

Double clicking on the MATLAB icon, present on the desktop, a window appears as shown in Figure 4.1.

The window consists of four panels as discussed below:

- **Current Folder** – This panel is used to access the files.
- **Command Window** – This panel is used to enter command at the command line. It consists of a prompt (>>), where commands can be entered.
- **Workspace** – This panel is used to explore data that is created or imported from files.

- **Command History** – This panel is used to view or rerun commands that are entered at the command line.

Commands can be issued at the command prompt. For example a variable can be created by just assigning a value to it. Variable 'x' can be created as follows:

>> x = 1;

To display the value of 'x', it can be simply typed at the command prompt as follows:

>> x

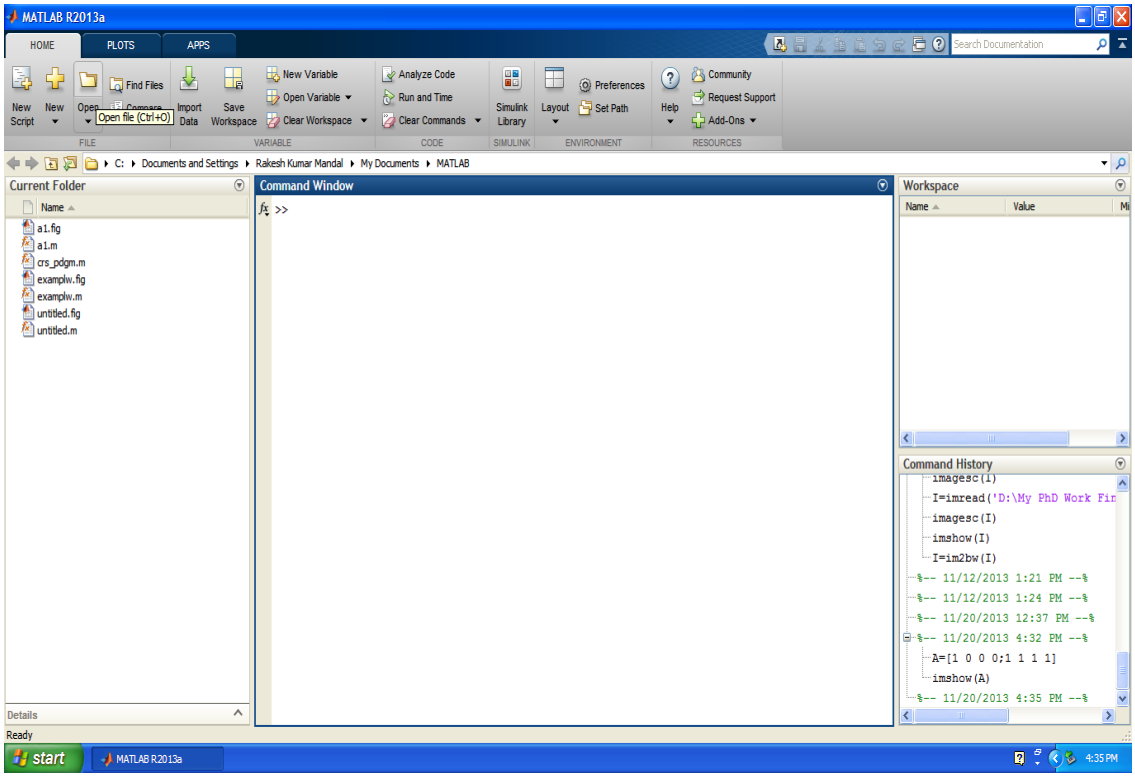

Figure 4.1: Opening Window of MATLAB

On pressing Enter button following value is displayed:

ans = 1

Similarly, some simple operations can be performed as follows:

Adding two numbers assigned to variables x and y, where the result is stored in another variable z.

>> x = 2;

```
>> y = 3;
>> z = x + y;
>> z
ans = 5
```

Multiplying two numbers assigned to variables x and y, where the result is stored in another variable z1.

```
>> z1 = x*y;
>> z1
ans = 6
```

The ‘tan’ of the value stored in variable z1 can calculated performing the following operation and the result can be stored in variable z2.

```
>> z2 = tan(z1);
>> z2
ans = -0.2910
```

Semicolon at the end of the command performs the computation but suppresses the display. Previous commands can be recalled by pressing up and down arrow keys either at an empty command line or after typing first few characters of a command.

4.2 Matrices and Arrays in MATLAB

Matrix Laboratory is abbreviated as MATLAB. MATLAB mainly performs operations on matrices and arrays. All the MATLAB variables are multidimensional arrays irrespective of the data type. A matrix is actually a two dimensional array [112]. Following are some examples of matrices with different dimensions in MATLAB.

A = [2] is an example of a matrix of order 1 x 1.

Similarly, B = [1 2] is an example of a matrix of order 1 x 2.

C = [1 2; 3 4] is an example of a matrix of order 2 x 2.

D = [1 2 3; 4 5 6; 7 8 9] is an example of a matrix of order 3 x 3.

A matrix of order 3 x 3 can be created as follows in MATLAB.

```
>> D = [1 2 3; 4 5 6; 7 8 9];
>> D
```

```
D =
1     2     3
4     5     6
7     8     9
```

Matrix can also be created by using functions like zeros, ones or rand. A matrix of order 3 x 3, with all ones, can be created as follows:

```
>> Z = ones (3, 3)
Z =
1     1     1
1     1     1
1     1     1
```

4.2.1 Matrix Operations

Some of the operations performed on matrices are discussed below.

4.2.1.1 Matrix Addition – Two matrices can be added as follows:

```
>> A = [1 2; 3 4];
>> B = [5 6; 7 8];
>> C = A + B;
>> C
C =
6     8
10    12
```

4.2.1.2 Matrix Multiplication – Two matrices can be multiplied as follows:

```
>> D=A*B;
>> D
D =
19    22
43    50
```

4.2.1.3 Transpose - A single quote (‘) is used to transpose a matrix.

```
>> D'
ans =
19    43
22    50
```

4.2.1.4 Concatenation – Two matrices can be joined to form a larger matrix. Concatenating matrices next to each other is called horizontal concatenation. For example,

```
>> E = [C, D]
E =
6     8    19    22
10    12    43    50
```

The pair of square brackets [] is the concatenation operator. In horizontal concatenation each matrix must have same number of rows. Similarly, matrices having same number of columns can be concatenated vertically using semicolons. For example,

```
>> F = [C; D]
F =
6     8
10    12
19    22
43    50
```

4.3 Programming in MATLAB

Programming in MATLAB is now becoming popular among researchers, scientists and engineers because it is an easy to learn software and saves time [112]. Any application can be programmed in MATLAB and saved with **‘.m’** extension. A file with **‘.m’** extension can be created in MATLAB as follows:

```
>> edit filename.m
```

Typing the above lines opens the editor window where a program can be written. The program written above can be saved as **'filename.m'** and executed as follows:

>>filename

4.3.1 Loops and Conditional Statements in MATLAB

MATLAB provides different looping and conditional statements. Some of them are 'for', 'while', 'if', 'switch' etc [112].

4.3.1.1 For Loop

To display first ten natural numbers using 'for' loop, the code is written as given below:

```
for i = 1:10
i
end
```

4.3.1.2 While Loop

To display first ten natural numbers using while loop, the code is written as given below:

```
i=1;
while (i<=10)
i
i = i + 1;
end
```

4.3.1.3 If Statement

'if' statement is used as follows:

```
i=4;
j=13;
if (i>j)
   'i is greater'
else
```

```
   'j is greater'
end
```

4.3.1.4 Continue

The continue statement passes control to the next iteration of the ‘for’ loop or ‘while’ loop, without executing the remaining statements in the body of the loop.

```
for i = 1:10
if (rem(i,2)==1)
continue
end
i
end
```

The above code displays only even numbers. The function ‘rem()’ is used to find out the remainder of a number.

4.3.1.5 Break

The break statement is used for an early exit from a ‘for’ loop or ‘while’ loop. It can also be used in ‘switch’ statements. Following example displays the searching of a particular element in an array. The control goes out of the loop using break after finding out the element.

```
I = [6 7 3 8 9 3];
J=8;
for i=1:6
   if (I(i) = = J)
      i
      break;
   end
end
```

4.3.1.6 Switch

The ‘switch’ statement executes different chunks of statements based on the value of a variable or expression. Following example shows a grading system using switch case statement.

```
percentage = 83;
n = percentage/10;
m=rem(percentage,10);
class=n-m/10;
class
switch (class)
   case 9
      'A+'
       break;
   case 8
      'A'
      break;
   case 7
      'B'
      break;
   case 6
      'C'
      break;
   case 5
      'D'
      break;
otherwise
   'Not Qualified'
end
```

4.4 Functions

Functions are the files in MATLAB that can accept input arguments and return output arguments. Different built-in functions, available in MATLAB, are used to perform different tasks in this work [150]. Some of the functions are discussed below:

4.4.1 imread() Function – This function is used to read an image file, stored in a particular location. The following statement reads a '.bmp' file stored in the given location and stores it in matrix A.

```
A=imread('D:\My PhD Work Final\PDGM\test samples1\a.bmp');
```

4.4.2 imwrite() Function – This function writes an image to a file in a specified format. The following statement writes the image A into the specified file stored in '.bmp' format.

```
imwrite(A,'c:\aa.bmp');
```

4.4.3 image() Function – This function displays an image with vertical and horizontal scales. The following statement displays the image A with vertical and horizontal scales.

```
image(A);
```

4.4.4 imshow() Function – This function is used to display an image without vertical and horizontal scales. The following statement displays the image A without vertical and horizontal scales.

```
imshow(A);
```

4.4.5 rgb2gray() Function – This function is used to convert an image file present in 'rgb' form into its equivalent grayscale form.

The following statement converts a 'rgb' file stored in matrix A into grayscale and stores the grayscale form in matrix A.

```
A=rgb2gray(A);
```

4.4.6 dither() Function – This function is used to convert an image file present in 'grayscale' form into its equivalent 'binary' form. The following statement converts a 'grayscale' file stored in matrix A into 'binary' and stores the 'binary' form in matrix A.

```
A=dither(A);
```

4.4.7 imresize() Function – This function is used to resize a matrix of any order into any other order. The following resizes the matrix A of a particular order into a dimension of order 80 x 80.

```
A = imresize(A,[80 80]);
```

4.4.8 rem() Function – This function is used to find out the remainder of a number. The following example finds out the remainder of a number and stores it in a variable.

```
k = rem(i,2);
```

4.5 Designing Interfaces in MATLAB

Interfaces can also be designed in MATLAB. The steps to create interfaces in MATLAB are given below:

STEP 1. Double click on the MATLAB icon given on the desktop.

STEP 2. Click on file menu.

STEP 3. A drop down menu will appear. Click on new.

STEP 4. A sub menu will appear. Click on 'GUI' menu.

STEP 5. A window will appear having two tabs. 'Create New GUI' and 'Open Existing GUI'.

STEPS 6. In order to open an existing GUI, click on the tab 'Open Existing GUI' and chose the file.

STEP 7. In order to design a new interface, on the window having 'Create New GUI' tab, which is already clicked and have four 'GUIDE template' options, where 'BLANK GUI (Default)' is already selected, click on ok.

STEP 8. Following window will appear with left and right panel. Left panel shows the tools required to design the interface and right panel is used to design the interface.

Figure 4.2 displays the Graphical User Interface of MATLAB.

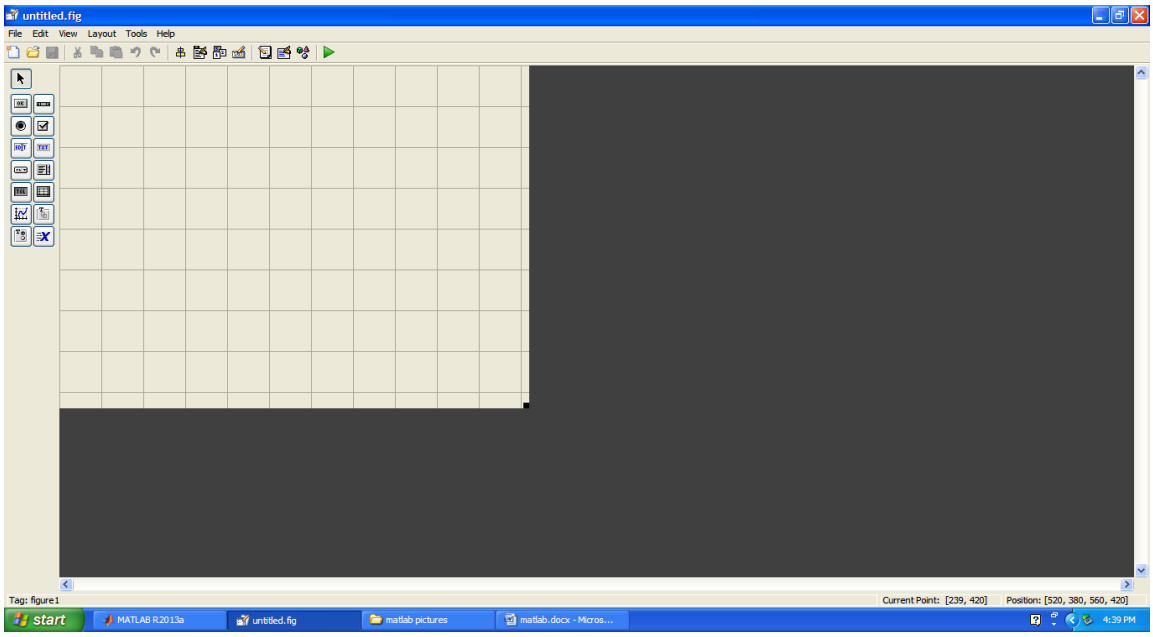

Figure 4.2: Graphical User Interface of MATLAB

4.6 Conclusion

MATLAB is an excellent programming language with rich sets of tools to carry out effective work. Different types of numerical computations including matrix manipulation can be done with ease using this software. MATLAB provides an interactive environment with a number of built-in functions. Different applications like data analysis, visualization as well as application development can be done using MATLAB.